

PATENT  
83394.0008

Express Mail Label No. EV 325 216 686 US

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re application of:

Hiromichi YAMADA et al.

Serial No: Not assigned

Filed: June 30, 2003

For: MICRO CONTROLLER FOR  
PROCESSING COMPRESSED CODES

Art Unit: Not assigned

Examiner: Not assigned

TRANSMITTAL OF PRIORITY DOCUMENT

Mail Stop PATENT APPLICATION

Commissioner for Patents

P.O. Box 1450

Alexandria, VA 22313-1450

Dear Sir:

Enclosed herewith is a certified copy of Japanese patent application No. 2002-192573 which was filed July 1, 2002, from which priority is claimed under 35 U.S.C. § 119 and Rule 55.

Acknowledgment of the priority document(s) is respectfully requested to ensure that the subject information appears on the printed patent.

Respectfully submitted,

HOGAN & HARTSON L.L.P.

By: 

Anthony J. Orler

Registration No. 41,232

Attorney for Applicant(s)

Date: June 30, 2003

500 South Grand Avenue, Suite 1900

Los Angeles, California 90071

Telephone: 213-337-6700

Facsimile: 213-337-6701

日 本 国 特 許 庁  
JAPAN PATENT OFFICE

別紙添付の書類に記載されている事項は下記の出願書類に記載されている事項と同一であることを証明する。

This is to certify that the annexed is a true copy of the following application as filed with this Office

出 願 年 月 日

Date of Application:

2002年 7月 1日

出 願 番 号

Application Number:

特願2002-192573

[ ST.10/C ]:

[ JP 2002-192573 ]

出 願 人

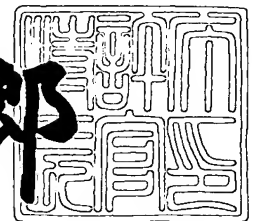
Applicant(s):

株式会社日立製作所

2003年 4月25日

特 許 庁 長 官  
Commissioner,  
Japan Patent Office

太田 信一郎



出証番号 出証特2003-3030606

【書類名】 特許願

【整理番号】 HA14471000

【提出日】 平成14年 7月 1日

【あて先】 特許庁長官 殿

【国際特許分類】 G06F 9/32  
G06F 9/38

【発明者】

【住所又は居所】 茨城県日立市大みか町七丁目 1 番 1 号 株式会社日立製作所 日立研究所内

【氏名】 山田 弘道

【発明者】

【住所又は居所】 茨城県日立市大みか町七丁目 1 番 1 号 株式会社日立製作所 日立研究所内

【氏名】 安部 雄一

【発明者】

【住所又は居所】 茨城県日立市大みか町七丁目 1 番 1 号 株式会社日立製作所 日立研究所内

【氏名】 中塚 康弘

【発明者】

【住所又は居所】 東京都小平市上水本町五丁目 2 0 番 1 号 株式会社日立製作所 半導体グループ内

【氏名】 山崎 尊永

【特許出願人】

【識別番号】 000005108

【氏名又は名称】 株式会社 日立製作所

【代理人】

【識別番号】 100084032

【弁理士】

【氏名又は名称】 三品 岩男

【電話番号】 045(316)3711  
【手数料の表示】  
【予納台帳番号】 011992  
【納付金額】 21,000円  
【提出物件の目録】  
【物件名】 明細書 1  
【物件名】 図面 1  
【物件名】 要約書 1  
【プルーフの要否】 要

【書類名】 明細書

【発明の名称】 圧縮コードを処理するマイクロコントローラ

【特許請求の範囲】

【請求項 1】

プログラムにしたがって処理を行なう CPU を備えたマイクロコントローラであって、

プログラムのコードを可変長コードに変換した圧縮コードと、

グループ化されたプログラムのコードについて、各グループの開始アドレスを特定するためのアドレス変換情報と、

各グループごとに、グループに含まれる各圧縮コードのコード長を特定するための圧縮コードタイプ情報とを記憶するメモリと、

CPU が出力するコードアドレスから、参照すべきアドレス変換情報と圧縮コードタイプ情報とを特定し、特定されたアドレス変換情報と圧縮コードタイプ情報とを用いて、対応する圧縮コードアドレスを求め、該当する圧縮コードを前記メモリから読み出す圧縮コード処理部と

を備えることを特徴とするマイクロコントローラ。

【請求項 2】

請求項 1 に記載のマイクロコントローラであって、

前記メモリは、圧縮コードを元のコードに伸張するための辞書情報をさらに記憶し、

前記圧縮コード処理部は、前記辞書情報を参照して、読み出した圧縮コードを元のコードに伸張することを特徴とするマイクロコントローラ。

【請求項 3】

請求項 1 に記載のマイクロコントローラであって、

前記圧縮コード処理部は、前記メモリ中の圧縮コードを記憶した領域と、アドレス変換情報を記憶した領域と、圧縮コードタイプ情報を記憶した領域とを識別するための情報を記憶していることを特徴とするマイクロコントローラ。

【請求項 4】

請求項 3 に記載のマイクロコントローラであって、

前記メモリは、プログラムのコードのブロックの順番に前記アドレス変換情報を記憶し、

プログラムのコードの順番に圧縮コードタイプ情報を記憶することを特徴とするマイクロコントローラ。

【請求項 5】

請求項 2 に記載のマイクロコントローラであって、

前記辞書情報は、対応する圧縮コードのコード長ごとに領域を分けて記憶され、それぞれの領域は、対応する圧縮コードの符号の順番に辞書情報が記憶されることを特徴とするマイクロコントローラ。

【請求項 6】

請求項 5 に記載のマイクロコントローラであって、

前記圧縮コード処理部は、圧縮コードタイプ情報から参照すべき辞書情報が記憶されている領域を特定し、圧縮コードをもとに、特定された領域に含まれる参照すべき辞書情報を特定することを特徴とするマイクロコントローラ。

【請求項 7】

請求項 1 に記載のマイクロコントローラであって、

前記圧縮コード処理部は、圧縮コードを読み出す際に、読み出すべき圧縮コードを含む所定のサイズの圧縮コード群を前記メモリから読み出すものであり、

直前に用いたアドレス変換情報と、圧縮コードタイプ情報と、圧縮コード群とをそれぞれ一時的に記憶する領域を備え、

CPU が出力したコードアドレスが、直前に読み出した圧縮コードと同一のブロックに含まれる場合には、前記領域に一時的に記憶しているアドレス変換情報と圧縮コードタイプ情報とを用い、

さらに、CPU が出力したコードアドレスに対応する圧縮コードが、直前に読み出した圧縮コード群に含まれる場合には、前記領域に一時的に記憶している圧縮コード群から圧縮コードを読み出すことを特徴とするマイクロコントローラ。

【請求項 8】

請求項 1 に記載のマイクロコントローラにおいて、

前記圧縮コードには、元のコードと同じコードが含まれることを特徴とするマ

イクロコントローラ。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】

本発明は、マイクロコントローラに係り、特に、メモリに記憶されている圧縮コードを処理するマイクロコントローラに関する。

【0002】

【従来の技術】

マイクロコントローラ（以下「マイコン」ともいう）は、家電製品、AV機器、携帯電話、自動車等の機器に組み込まれ、メモリに記憶されているプログラムコードにしたがって処理を行なうことで、それぞれの機器の制御を行なう装置である。

【0003】

マイコンは、機器に組み込まれて利用されるというその性質上、機器の制御を行なうために必要な性能を備え、かつ、安価であることが要求される。

【0004】

ところが、近年、マイコンが行なう処理は複雑化しており、処理に必要なプログラムの容量は増加している。このため、マイコンに占めるプログラムメモリの比率が高くなっており、この傾向は今後も続くと考えられている。一般に、プログラムメモリの容量が増えるとその分コストが上昇するため、安価なマイコンを提供するためには、プログラムメモリの容量をいかに抑えるかが重要な問題となっている。

【0005】

ところで、携帯情報端末等の汎用型の情報処理装置では、プログラムメモリの容量を抑えるために、プログラムコードを圧縮する技術が提案され、実装化も進んでいる。

【0006】

例えば、IBM社が提唱しているCodePackと呼ばれるコード圧縮技術は、元のコードを圧縮してプログラムメモリ等に格納する。このとき、圧縮率を

高めるために、圧縮コードは出現頻度等に応じた可変長のビットパターンとしておく。そして、プログラムメモリ等からの読み出し時に、所定のブロック単位で伸長して、伸長後の元のコードをブロック単位でキャッシュメモリに格納するようにしている。

#### 【 0 0 0 7 】

圧縮コードは可変長のため、圧縮後の各コードのメモリアドレスは不規則になる。このため、圧縮コード中に自身のコード長に関する情報を含めるようにしておき、読み出し時には、この情報を元に、各圧縮コードのメモリアドレスを特定して、ブロックの先頭の圧縮コードから順番に伸長するようにしている。

#### 【 0 0 0 8 】

このコード圧縮技術は、ブロック単位で圧縮コードを伸長してキャッシュメモリに格納することから、キャッシュメモリのヒット率が高く、また、プログラムの平均的な実行性能を高くすることが重要とされる汎用の情報処理装置において特に有効と考えられる。

#### 【 0 0 0 9 】

##### 【発明が解決しようとする課題】

しかしながら、従来提唱されているコード圧縮技術は、上述のように、キャッシュメモリの高いヒット率が期待でき、プログラムの平均的な実行性能を高くすることが重要な汎用の情報処理装置には適していても、マイコン等の組み込み機器には、必ずしも最適であるとはいえないと考えられる。

#### 【 0 0 1 0 】

これは以下のような理由による。すなわち、マイコン等の組み込み機器では、一般に、プログラムのループ処理が少なく、キャッシュメモリのヒット率は、汎用の情報処理装置ほど高くはならない。このため、キャッシュメモリの書き換えが頻繁に発生することになる。キャッシュメモリの書き換えはブロック単位で行なわれることから、書き換えのたびに、目的のコードを含むブロックの伸長が行なわれる。このとき、目的コード以外のコードは、マイコン等の組み込み機器では、必ずしも直後に使用されとは限らないため、目的のコードを取得するためのだけの余分な伸長処理となりかねず、コード伸張時のオーバーヘッドが大きくなるから



である。また、組み込み機器では、高いヒット率は期待できないことから、そもそも、キャッシュメモリを備えない場合もあるからである。さらに、マイコン等の組み込み機器では、プログラムの平均的な実行性能よりも、制御周期内に決められた処理を確実に実行できるというリアルタイム性が重視されることが多いからである。

【 0 0 1 1 】

したがって、マイコン等の組み込み機器では、汎用型の情報処理装置に比べ、目的とする圧縮コードのアドレスをより高速に特定できることが重要となってくる。

【 0 0 1 2 】

本発明の目的は、マイコン等の組み込み機器に好適なコード圧縮技術を提供することにある。

【 0 0 1 3 】

【課題を解決するための手段】

上記課題を解決するため、本発明によれば、

プログラムにしたがって処理を行なうCPUを備えたマイクロコントローラであって、

プログラムのコードを可変長コードに変換した圧縮コードと、

グループ化されたプログラムのコードについて、各グループの開始アドレスを特定するためのアドレス変換情報と、

各グループごとに、グループに含まれる各圧縮コードのコード長を特定するための圧縮コードタイプ情報とを記憶するメモリと、

CPUが出力するコードアドレスから、参照すべきアドレス変換情報と圧縮コードタイプ情報とを特定し、特定されたアドレス変換情報と圧縮コードタイプ情報とを用いて、対応する圧縮コードアドレスを求め、該当する圧縮コードを前記メモリから読み出す圧縮コード処理部と

を備えることを特徴とするマイクロコントローラが提供される。

【 0 0 1 4 】

ここで、前記メモリは、圧縮コードを元のコードに伸張するための辞書情報を

さらに記憶し、

前記圧縮コード処理部は、前記辞書情報を参照して、読み出した圧縮コードを元のコードに伸張することができる。

#### 【作用】

プログラムのコードを可変長コードに変換した圧縮コードと、プログラムのコードをグループ化し、変換後の各グループの開始アドレスを特定するためのアドレス変換情報と、各グループごとに、変換後のグループに含まれる各圧縮コードのコード長を特定するための圧縮コードタイプ情報とをメモリに記憶し、CPUが出力するコードアドレスから対応する圧縮コードアドレスを直接計算できるように構成することで、マイコン等の組み込み機器に好適なコード圧縮が達成される。

#### 【0015】

#### 【発明の実施の形態】

本発明の実施の形態について図面を参照して説明する。

#### 【0016】

図1は、本発明を適用したマイクロコントローラ（マイコン）の実施例の構成の要部を説明するためのブロック図である。

#### 【0017】

マイコンは、プログラムを実行するCPU1と、プログラムをあらかじめ圧縮して記憶する領域を含むメモリ2と、CPU1からの命令フェッチまたはデータ読み出しの要求に対し、メモリ2から圧縮コードを読み出して元のコードに伸長し、これをCPU1に供給する圧縮コード伸長回路3とを備えて構成される。

#### 【0018】

メモリ2は、プログラムのコードを1種以上のコード長の符号に変換した圧縮コードを記憶する圧縮コード領域23と、CPU1が出力するコードアドレスをこれに対応する圧縮コードアドレスに変換するためのアドレス変換情報および圧縮コードタイプ情報をそれぞれ記憶するアドレス変換情報領域21および圧縮コードタイプ情報領域22と、圧縮コードを元のコードに変換するための辞書を記憶する辞書領域24とを含んで構成されている。

## 【 0 0 1 9 】

CPU1は、CPUアドレスバス1000にコードアドレスを出力し、CPUデータバス1001より伸長されたコードを読み込む。圧縮コード伸長回路3は、メモリバス3000にアドレスを出力してメモリ2をアクセスし、メモリデータバス3001より、アドレス変換情報、圧縮コードタイプ情報、圧縮コード、辞書を必要に応じて読み出し、CPUデータバス1001に伸長したコードを出力する。

## 【 0 0 2 0 】

図2は、本発明の基本的な原理を示す第1の実施例における、圧縮コードのフォーマットを説明するための図である。ここでは、4種類のフォーマットの例で説明するが、本発明においては、圧縮コードの種類はこれに限定されるわけではない。また、圧縮する元のコードの単位を16ビットとして説明を行うが、本発明においてはこれに限定されるわけではない。現状、マイコンで処理される命令やデータのサイズとしては、8ビット、16ビット、32ビットが一般的であることから、本実施例における圧縮する元のコードの単位を16ビットとした。

## 【 0 0 2 1 】

図2(a)は、コード長が4ビットのタイプ0フォーマットを示す。2進表記で0000から1111までの16種類の圧縮コードを定義でき、16ビットコードを4ビットへと25%に圧縮することができる。

## 【 0 0 2 2 】

図2(b)は、コード長が8ビットのタイプ1フォーマットを示す。2進表記で00000000から11111111までの256種類の圧縮コードを定義でき、16ビットコードを8ビットへと50%に圧縮することができる。

## 【 0 0 2 3 】

図2(c)は、コード長が10ビットのタイプ2フォーマットを示す。2進表記で0000000000から1111111111までの1024種類の圧縮コードを定義でき、16ビットコードを10ビットへと62.5%に圧縮することができる。

## 【 0 0 2 4 】

図 2 (d) は、コード長が 1 6 ビットのタイプ 3 フォーマットを示す。このフォーマットは、タイプ 0、1、2 と異なり、圧縮前のコードをそのまま 1 6 ビットフィールドに持つ。このフォーマットではコードの圧縮は行われない。

#### 【 0 0 2 5 】

タイプ 0、1、2 のフォーマットの圧縮コードに対しては、これを元のコードに伸長するために辞書を参照することが必要である。

#### 【 0 0 2 6 】

圧縮率を最も高くするという目的のためには、プログラムごとにコードの出現頻度を調べ、出現頻度の高いものから、タイプ 0、タイプ 1、タイプ 2、タイプ 3 の順に圧縮コードを定義するのが良い。しかしながら、タイプ 3 フォーマットは辞書を引用する必要がないため、他のタイプに比べて伸長を高速に行える。したがって、出現頻度が高いコードであっても、伸長を高速に行うために、タイプ 3 フォーマットに割り当てるということも考えられ、本発明においてはコードの出現頻度とタイプの割り当てについて、特に制限を設けるものではない。

#### 【 0 0 2 7 】

図 3 は、圧縮前のコードと圧縮コードとの対応を説明するための図である。図 3 (a) は、圧縮前のコードのメモリマップの一部を示す図である。図中、ブロック I、ブロック J、ブロック K は、圧縮前コードのメモリマップを 1 6 ビット  $\times$  1 6 の 2 5 6 ビット (3 2 バイト) 単位に分割した領域の一部である。ブロック J の領域に示す a ~ p は、それぞれ 1 6 ビットのコードで、ブロックの先頭からの順番を区別するためにアルファベット文字を割り当てたものであり、コードの値とは何ら関係は無い。また、本実施例では、1 ブロックに含まれるコードの数を 1 6 として説明するが、本発明においては、この数を 1 6 に限定するものではない。ただし、後述するアドレス変換情報や圧縮コードタイプ情報のアドレス計算を容易にできるなどの理由のために、2 のべき乗の数であるほうが望ましい。

#### 【 0 0 2 8 】

図 3 (b) は、ブロック I、ブロック J、ブロック K に対応する圧縮コード領域 2 3 のメモリマップを示す。圧縮コードの各ブロックは、1 6 ビット境界から

コードが配置されている。ブロック J の領域については、圧縮コード a ～ p の配置を示している。圧縮コードは、図 2 のフォーマットで説明したように、4 ビット、8 ビット、1 0 ビット、1 6 ビットのいずれかのサイズとなる。例えば、ブロック J の先頭の 1 6 ビットには圧縮コード a、b、c が収まっており、続く 1 6 ビットには圧縮コード d と e の一部が収まっている。このように、ブロックの先頭から圧縮コードを 1 6 個連続して配置するため、最後の圧縮コード p の後には隙間（コードが無い部分）が生じる場合がある。

## 【 0 0 2 9 】

図 3（a）圧縮前コードのアドレスと図 3（b）圧縮コードのアドレスとの対応は、コード単位ではなく、ブロック単位で管理する。すなわち、圧縮コードの各ブロックの先頭アドレスを保持するようにする。図 3 では、圧縮前のブロック I、ブロック J、ブロック K について、それぞれの圧縮後のブロックの先頭のアドレスに変換される様子を示している。このブロック単位のアドレス変換情報（圧縮後のブロックの先頭アドレスを示す情報）は、図 1 に示したようにメモリ 2 上のアドレス変換情報領域 2 1 に記憶しておく。

## 【 0 0 3 0 】

図 4 は、圧縮前コードのブロックとアドレス変換情報との対応を説明するための図である。ここで、圧縮の対象とするコードの容量を最大 8 メガバイトとして説明をする。図 4（a）は、圧縮前のコード（最大 8 メガバイト）のメモリマップを示す。図 3 で説明したように、圧縮前コードの 1 ブロックには、1 6 ビットコードが 1 6 個の 3 2 バイトが含まれるため、ブロックの数は最大で 2 6 2 1 4 4 となる。

## 【 0 0 3 1 】

図 4（b）は、アドレス変換情報領域 2 1 のメモリマップを示す。アドレス変換情報は、1 ブロックにつき 3 2 ビット（4 バイト）の情報を持つものとする。アドレス変換情報領域 2 1 は、メモリ 2 上のアドレス変換情報ベースアドレスと呼ぶアドレスから、ブロックの順番にアドレス変換情報を連続して記憶するものとする。アドレス変換情報領域 2 1 は最大で、4 バイト × 2 6 2 1 4 4 の 1 メガバイトとなる。

## 【0032】

図5は、アドレス変換情報のメモリアドレスの計算方法を示す図である。すなわち、CPUアドレスから、参照すべきアドレス変換情報のアドレスを求めるための計算方法である。ここで、CPUアドレスは、CPU1が出力するコードアドレスである。CPU1は、圧縮前のコードアドレスをそのまま出力する。アドレス長は32ビットで、単位はバイトであるとし、最大で4ギガバイトのアドレス空間を指定することができる。

## 【0033】

本実施形態では、メモリ2上の任意の8メガバイト境界からはじまる8メガバイトの領域を、圧縮コード領域23として設定するようになる。このため、CPUアドレスの10ビット目から続く18ビット（A22-A5フィールド）は、コードのブロック番号を示し、最後の5ビット（A4-A0フィールド）は、ブロック内のバイトアドレスを示すことになる。

## 【0034】

アドレス変換情報ベースアドレスは、上位12ビット（B31-B20フィールド）のみ値を設定可能で、下位20ビットは全て0固定とし、メモリ2上の1メガバイト境界を指定できるものとする。アドレス変換情報ベースアドレスは、圧縮コード伸長回路3内のレジスタ等に記憶しておく。

## 【0035】

アドレス変換情報のメモリアドレスは、アドレス変換情報ベースアドレスのB31-B20フィールドと、CPUアドレスのA22-A5フィールドを使い、本図に示す方法で生成する。すなわち、上位12ビットをアドレス変換情報ベースアドレスのB31-B20フィールドとし、続く18ビットをCPUアドレスのA22-A5フィールドとし、最後の2ビットを0とする。

## 【0036】

次に、図1のメモリ2の圧縮コードタイプ情報領域22に記憶される圧縮コードタイプ情報について説明する。圧縮コードタイプ情報は、図2で説明した圧縮コードのフォーマットのタイプを識別する情報である。本実施例では、4種類のタイプを識別するために、2ビットのコードを使い、タイプ0は“00”、タイ

ブ1は“01”、タイプ2は“10”、タイプ3は“11”とそれぞれ定義するものとする。このように、本実施形態では、圧縮コードのタイプ情報を、圧縮コードとは別に管理するようにしている。

#### 【0037】

図6は、圧縮コード1ブロックに対する圧縮コードタイプ情報のフォーマットを示す図である。図3と同様に、ブロックの先頭から16の圧縮コードにa～pと名前を付けると、圧縮コードタイプ情報は、32ビットの上位から、a～pというように、圧縮コードと同じ順番に圧縮コードタイプの識別情報を配置する。

#### 【0038】

図7は、圧縮前コードと圧縮コードタイプ情報との対応を説明するための図である。図7(a)は、圧縮前のコード（最大8メガバイト）の仮想メモリマップを示す。圧縮前コードの1ブロックには、16ビットコードが16個の32バイトが含まれるため、ブロックの数は最大で262144となる。

#### 【0039】

図7(b)は、圧縮コードタイプ情報領域22のメモリマップを示す図である。圧縮コードタイプ情報は、1ブロックにつき32ビット（4バイト）の情報を持つ。圧縮コードタイプ情報領域22は、メモリ2上の圧縮コードタイプ情報ベースアドレスと呼ぶアドレスから、ブロックの順番に圧縮コードタイプ情報を連続して記憶する。圧縮コードタイプ情報領域22は最大で、4バイト×262144の1メガバイトとなる。

#### 【0040】

図8は、圧縮コードタイプ情報のメモリアドレスの計算方法を示す図である。前述のように、CPUアドレスのA22-A5フィールドは、コードのブロック番号を示す。

#### 【0041】

圧縮コードタイプ情報ベースアドレスは、上位12ビット（B31-B20フィールド）のみ値を設定可能で、下位20ビットは全て0固定とし、メモリ2上の1メガバイト境界を指定できるものとする。

#### 【0042】

圧縮コードタイプ情報のメモリアドレスは、アドレス変換情報ベースアドレスの B 3 1 - B 2 0 フィールドと、CPU アドレスの A 2 2 - A 5 フィールドを使い、本図に示す方法で生成する。すなわち、上位 1 2 ビットを圧縮コードタイプ情報ベースアドレスの B 3 1 - B 2 0 フィールドとし、続く 1 8 ビットを CPU アドレスの A 2 2 - A 5 フィールドとし、最後の 2 ビットを 0 とする。

## 【 0 0 4 3 】

次に、図 1 のメモリ 2 の辞書領域 2 4 に記憶される辞書について説明する。辞書は、図 2 で説明した圧縮コードを元のコードに変換するためのテーブルである。コード長が 4 ビットであるタイプ 0 フォーマットの辞書は 1 6 ビット × 1 6、コード長が 8 ビットであるタイプ 1 フォーマットの辞書は 1 6 ビット × 2 5 6、コード長が 1 0 ビットであるタイプ 2 フォーマットの辞書は 1 6 ビット × 1 0 2 4 のテーブルとなる。

## 【 0 0 4 4 】

図 9 は、辞書領域 2 4 のメモリマップを示す図である。辞書領域 2 4 は、メモリ 2 上の辞書ベースアドレスと呼ぶアドレスから、タイプ 2、タイプ 1、タイプ 0 の順番に辞書を記憶するものとする。

## 【 0 0 4 5 】

図 1 0 は、参照すべき辞書のメモリアドレスの計算方法を示す図である。図 1 0 ( a ) に示すように辞書ベースアドレスは、下位 1 2 ビットは全て 0 固定とし、メモリ上の 4 0 9 6 バイト境界を指定できるものとする。図 1 0 ( b )、図 1 0 ( c )、図 1 0 ( d ) はそれぞれ、タイプ 2、タイプ 1、タイプ 0 の辞書アドレスである。辞書領域 2 4 には、それぞれのタイプごとに、圧縮コードの順番（タイプ 0 を例にすると、0 0 0 0、0 0 0 1、0 0 1 0・・・の順）に元のコードが格納されている。このため、辞書のメモリアドレスは、辞書ベースアドレスの A 3 1 - A 1 2 フィールドと、タイプごとのオフセット値と、圧縮コードと、再開ビットの「0」とを結合することによって生成される。ここで、タイプごとのオフセット値は、タイプ 2 は「0」であり、タイプ 1 は「1 0 0」であり、タイプ 0 は、「1 0 1 0 0 0 0」となる。

## 【 0 0 4 6 】



次に、圧縮コード伸長回路 3 が行なう圧縮コードアドレスの計算処理について説明する。図 3 に戻って、CPU 1 がブロック J の先頭から 8 番目のコード h を読み出すという例で説明する。まず、CPU 1 が出力したコードアドレスから図 5 で示したアドレス計算を行い、ブロック J のアドレス変換情報を読み出す。次に、図 8 で示したアドレス計算を行い、ブロック J の圧縮コードタイプ情報を読み出す。圧縮コードタイプ情報により、ブロック内の 16 の圧縮コードのそれぞれのサイズを知ることができる。すなわち、タイプ 0 は 4 ビット、タイプ 1 は 8 ビット、タイプ 2 は 10 ビット、タイプ 3 は 16 ビットである。ブロック J の先頭から 8 番目の圧縮コード h までの距離（ビット数）は、ブロック J の圧縮コードの 1 番目から 7 番目まで（a ~ g）のサイズを加算したものに等しい。

【0047】

図 11 は、ブロックの先頭から、目的の圧縮コードまでのビット数を計算する回路の例を示す図である。図中 a ~ o は、ブロックの先頭から 15 番目までの圧縮コードのタイプ情報であり、それぞれのタイプ情報を回路 330 において、圧縮コードサイズ  $s_a \sim s_o$  に変換する。

【0048】

図 12 は、回路 330 の動作を示す図である。回路 330 は圧縮コードのタイプ情報  $i$ （2 ビット）をサイズ情報  $o$ （4 ビット）に変換する。サイズ情報  $o$  の値は、入力  $i$  が示すタイプのコード長を 2 で割った値である。すなわち、タイプ 0 は 2、タイプ 1 は 4、タイプ 2 は 5、タイプ 3 は 8 となる。これは、圧縮コードのサイズが 2 ビットの整数倍であるため、あらかじめ、2 で割った（下位方向に 1 ビットシフトした）データとしておくことにより、図 11 の回路全体の規模を小さく抑えることができるからである。

【0049】

図 11 に示した回路 331 は、圧縮コードサイズ  $s_a \sim s_o$  と、CPU アドレスのビット 4 - 1（ $A[4:1]$ ）を入力し、圧縮コードサイズのうち、目的の圧縮コードに対応する部分とこれ以降のデータとを 0 にマスクする。 $m_a \sim m_o$  をマスク後の圧縮コードサイズと呼ぶことにする。

【0050】

図 1 3 は、回路 3 3 1 の動作を示す図である。CPU アドレスの  $A[4:1]$  と、マスク後の圧縮コードサイズ  $ma \sim mo$  の値を示している。ここで、 $Z$  は 4 ビット全て 0 を意味する。ブロックの 8 番目のコード  $h$  の場合は、 $A[4:1] = 0111$  となり、 $ma \sim mg$  はそれぞれ  $sa \sim sg$ 、 $mh \sim mo$  は  $Z$  となる。

【0051】

図 1 1 に示した回路 3 3 2 は、マスク後の圧縮コードサイズ  $ma \sim mo$  を加算する回路である。この回路の出力が、ブロックの先頭から目的の圧縮コードまでのブロック内アドレスとなる。

【0052】

図 1 4 は、回路 3 3 2 の例を示す図である。マスク後の圧縮コードサイズ  $ma \sim mo$  を桁上げ保存加算アレイ (CSA アレイ) 3 3 2 - 1 で加算し、桁上げ保存形式の出力  $s$ 、 $r$  を桁上げ伝播加算器 (CPA) 3 3 2 - 2 で加算し、ブロック内アドレス (単位は 2 ビット) を出力する。

【0053】

図 1 5 は、CSA アレイ 3 3 2 - 1 の構成例を示す図である。回路 3 3 2 - 1 0 は、3 入力 2 出力の桁上げ保存加算器であり、その動作を図 1 6 に示す。 $i_0$ 、 $i_1$ 、 $i_2$  は、それぞれ 1 ビットの入力で、 $r$  は桁上がり (キャリー) 出力、 $s$  は和 (サム) 出力である。図 1 5 において、マスク後の圧縮コードサイズは、1 つのコード (例えば  $a$ ) に対して 4 ビットで表され、 $ma_3$ 、 $ma_2$ 、 $ma_1$ 、 $ma_0$  のように表されている。図の上段から下段に向かって、各段ごとに 1 コード分のマスク後の圧縮コードサイズを入力し、出力を次の段に入力する構成となっている。また、図のなかで、桁上げ保存加算回路 3 3 2 - 1 0 の入力が 0 となっているのは、データ 0 を意味する。CSA アレイ 3 3 2 - 1 の出力は 7 ビットのサム  $s_6 \sim s_0$  と 6 ビットのキャリー  $r_5 \sim r_0$  となる。図 1 4 の CPA 3 3 2 - 2 で、 $s_6 \sim s_0$  と、 $r_5 \sim r_0$  を 1 ビット上位にシフトしたデータの加算を行うことにより、ブロック内アドレスが計算される。

【0054】

上記の構成において、圧縮コード伸長回路 3 が、CPU 1 が出力したコードアドレスに対応する圧縮コードのアドレスを特定し、伸長する処理の原理について

説明する。

【 0 0 5 5 】

圧縮コード伸長回路 3 は、CPU 1 が出力したコードアドレスに対して、図 5 に示した処理によりアドレス変換情報のアドレスを求めて、メモリ 2 のそのアドレスを参照することで、目的とする圧縮コードを含むブロックの先頭アドレスを取得する。

【 0 0 5 6 】

そして、図 8 に示した処理により、圧縮コードタイプ情報のアドレスを求めて、メモリ 2 のそのアドレスを参照することで、目的とする圧縮コードを含むブロックの圧縮コードタイプ情報を取得する。

【 0 0 5 7 】

取得した圧縮コードタイプ情報から、図 1 1 に示した回路により、目的とする圧縮コードのブロック内におけるアドレスが求められるため、取得したブロックの先頭アドレスを加算することで、目的とする圧縮コードのアドレスを求めることができる。また、圧縮コードタイプ情報から、目的とする圧縮コードの圧縮形式のタイプ（コード長）がわかるため、メモリ 2 から目的とする圧縮コードを取得することができる。

【 0 0 5 8 】

そして、圧縮形式のタイプにより伸長が必要のない場合には、取得したコードをそのまま CPU 1 に出力する。一方、伸長が必要な場合には、図 1 0 に示した処理により、辞書のアドレスを求めて、メモリ 2 のそのアドレスを参照することで、CPU 1 が要求するコードを取得することができる。そして、このコードを CPU 1 に出力する。

【 0 0 5 9 】

このように、本実施形態によれば、目的とする圧縮コードのアドレスを直接指定して、その圧縮コードのみを伸長することにより、マイコン等の組み込み機器に好適な、高い圧縮率と、高速な伸長性能とを実現している。

【 0 0 6 0 】

次に、実装上、圧縮コード伸長回路 3 が目的の圧縮コードを含む圧縮コード群

をメモリ 2 から読み出す処理について説明する。ここで、圧縮コード群は、メモリ 2 から読み出すデータの単位（例えば、4 バイト分のデータ）である。図 1 7 は、圧縮コードアドレスの計算方法を説明するための図である。図 1 7 (a) は、CPU 1 が読み出しを行うコードのアドレス変換情報、すなわち、目的の圧縮コードを含むブロックの先頭アドレス  $BA[31:0]$ （単位はバイト）である。図 1 7 (b) は、目的の圧縮コードの、ブロック内アドレス  $IA[6:0]$ （単位は 2 ビット）である。図 1 7 (c) は、目的の圧縮コードのアドレス  $CA[33:0]$ （単位は 2 ビット）の計算方法を示している。図に示すように、 $BA$  に対し、 $IA$  を下位に 2 ビットシフトして加算を行う。結果アドレスの上位 30 ビット  $CA[33:4]$  が、目的の圧縮コードを含む 4 バイトデータのメモリアドレスを表し、下位 4 ビット  $CA[3:0]$  が、メモリより読み出した 4 バイトデータにおける目的の圧縮コードの位置を表す。

## 【 0 0 6 1 】

図 1 8 は、圧縮コード伸長回路 3 が、メモリ 2 より読み出した圧縮コード群から、目的の圧縮コードを抽出する処理を説明するための図である。図 1 7 の  $CA[33:4]$  が示すアドレス（単位は 4 バイト）から読み出した圧縮コード群において、 $CA[3:0]$  が示すビット位置（単位は 2 ビット）から、圧縮コードのサイズ分を取り出すことにより、目的の圧縮コード  $h$  が得られる。

## 【 0 0 6 2 】

ここで、メモリ 2 より読み出した圧縮コード群に、目的の圧縮コードが一部しか含まれていない場合が発生する。例えば、図 1 8 において、圧縮コード  $i$  は、コードの先頭 2 ビットであり、完全なコードではない。これは、圧縮コード  $i$  のサイズ情報を使うことで検出可能である。この場合には、メモリ 2 上の次のアドレスにある 4 バイト分の圧縮コード群を読み出すことにより、目的の圧縮コードを得ることができる。

## 【 0 0 6 3 】

次に、圧縮コードの伸長に必要なメモリアクセスを効率的に行う方法について説明する。

## 【 0 0 6 4 】

図 1 9 は、圧縮コードの伸長の処理の流れを説明するためのフロー図である。  
 なお、圧縮コード伸張回路 3 は、直前に読み出したアドレス変換情報と圧縮コードタイプ情報と圧縮コードアドレスと圧縮コード群とをそれぞれ記憶するレジスタを備えているものとする。

## 【 0 0 6 5 】

まず、圧縮コード伸張回路 3 は、CPU 1 が出力するコードアドレスに対し、前回と同じブロックであるかを確認する（S 1 0 1）。同じブロックであれば、アドレス変換情報と圧縮コードタイプ情報とは、既にメモリ 2 から読み出し済みであるため次のステップへ移動する。ブロックが異なっていれば、アドレス変換情報と圧縮コードタイプ情報とをメモリ 2 から読み出す（S 1 0 2、S 1 0 3）。

## 【 0 0 6 6 】

次に、前回読み出した圧縮コード群に目的の圧縮コードが完全に含まれるかどうかを確認する（S 1 0 4）。含まれていれば次のステップへ移動し、含まれていなければ次の圧縮コード群をメモリ 2 から読み出す（S 1 0 5）。このとき、読み出した 4 バイトの圧縮コード群に目的の圧縮コードが完全に含まれていない場合には、さらに次のアドレスの圧縮コード群も読み出す。

## 【 0 0 6 7 】

次に、メモリ 2 から読み出した圧縮コード群から目的の圧縮コード（S 1 0 6）を抽出する。

## 【 0 0 6 8 】

そして、圧縮コードが非圧縮形式かどうかを確認する（S 1 0 7）。非圧縮形式の場合は、コードをそのまま CPU 1 に出力する。圧縮形式の場合には、辞書アドレスを計算し、メモリ 2 より辞書を読み出して元のコードに伸長してから（S 1 0 8）、CPU 1 にコードを出力する。

## 【 0 0 6 9 】

次に、本発明の第 2 の実施例について説明する。

## 【 0 0 7 0 】

図 2 0 は、第 1 の実施例ではメモリ 2 に記憶していた辞書を、専用の記憶装置

4に記憶するようにし、圧縮コード伸長回路3は、記憶装置4専用のアドレスバス4000とデータバス4001とを使用して、辞書をアクセスする構成を説明するためのブロック図である。例えば、メモリ2が低速、大容量メモリであるとし、メモリ2のアクセスにCPU1が動作するクロックの数サイクル分を要するとした場合に、小容量で高速な記憶装置4を辞書として使用することにより、圧縮コードの伸長に要する時間を短縮できる。命令フェッチの場合には、分岐や割り込みがない限り、連続したアドレスの命令がフェッチされることになる。アドレス変換情報21、圧縮コードタイプ情報22、圧縮コード23は、メモリから読み出されるデータに、複数コード分の情報が含まれるため、1命令のフェッチに対し、毎回読み出しを行うわけではない。しかし、辞書については、タイプ3以外のフォーマットの圧縮コードの伸長のために、多くの読み出しが必要になる。従って、容量が少なくて済む辞書を、小容量だが高速な記憶装置4に記憶することは、性能の面で有効であり、またハードウェアが多少増加するとしても、これよりはるかに大容量のメモリの削減効果を考えれば、十分実現性があると思われる。

#### 【0071】

図21は、本発明の第3の実施例を示すブロック図である。第2の実施例に対して、辞書専用の記憶装置が、辞書1用の記憶装置4と、辞書2用の記憶装置5との2つになり、それぞれが独立したバスで圧縮コード伸長回路3と接続される。この構成は、圧縮の単位に対して、CPUデータバス1001の幅が大きい場合に有効になる。例えば、CPUデータバス1001が32ビットで、圧縮の単位が16ビットの場合を考える。圧縮コード伸長回路3が辞書を1式のみ持つ場合は、圧縮形式であるタイプ0、1、2のフォーマットの圧縮コードは、1サイクルに1つしか伸長できない。タイプ3フォーマットは辞書を引用する必要がないため、他のコードといっしょにCPUデータバス1001に出力するようにできる。辞書を2式使用すれば、1サイクルに2つの伸長が可能になるため、より伸長の性能が向上することになる。

#### 【0072】

図22は、本発明の第4の実施例を示すブロック図である。ここで、BSC6

はバスステートコントローラで、CPU 1 の外部へのアクセスを制御する回路である。7 はマイコンの階層を表しており、その内部は本発明に関わる主要部のみを示している。外部メモリ 8 は、マイコン 7 と外部で接続されるメモリである。この構成では、圧縮コードを外部メモリ 8 にも記憶する。圧縮コードはメモリ 2 にも記憶されているが、内蔵メモリ 2 と外部メモリ 8 の両方に記憶することも可能であり、外部メモリ 8 のみに記憶することも可能である。マイコンの内蔵メモリ 2 だけでは容量が十分でない場合に、外部メモリ 8 を接続して使用することが考えられるが、外部メモリに圧縮コード 8 3 を記憶することにより、未圧縮のコードを記憶する場合に比べて、性能、消費電力を低減できる可能性がある。それは、マイコン 7 に読み込まれるコード量が、圧縮形式の方が少なくなるためである。特に、マイコン内部に比べて、外部バスの速度が遅い場合に効果が大きくなる。

## 【 0 0 7 3 】

次に、圧縮の方法について説明する。図 2 3 は、マイコンプログラムの圧縮の流れを示す図である。圧縮はソースコード 9 0 4、オブジェクトコード 9 0 5 の両方に対して行うことができる。プログラム開発環境 9 0 0 には、コンパイラ 9 0 1、リンカ 9 0 2、圧縮ツール 9 0 3 が含まれる。圧縮・非圧縮指定情報 9 0 6 とは、ソースコード 9 0 4 とオブジェクトコード 9 0 5 のどの部分を圧縮し、どの部分を非圧縮にするかの情報である。初期化ルーチン 9 0 7 は、マイコンのパワーオンリセット後に、圧縮コードの伸長に必要な情報、例えばアドレス変換情報ベースアドレス、圧縮コードサイズ情報ベースアドレスなどをレジスタ設定するプログラムである。コンパイラ 9 0 1、リンカ 9 0 2、圧縮ツール 9 0 3 の処理により、5 種類のデータが出力される。圧縮・非圧縮指定情報 9 0 6 で、非圧縮に指定した部分は、非圧縮コード 9 0 8 として出力される。圧縮に指定した部分は、アドレス変換情報 9 0 9、圧縮コードタイプ情報 9 1 0、圧縮コード 9 1 1、辞書 9 1 2 を生成する。これらの出力データをマイコンのメモリに書き込むことにより、非圧縮コード、圧縮コードを含むプログラムの実行が可能になる。

## 【 0 0 7 4 】

図 2 4 は、非圧縮コードと圧縮コードを混在させる方法を説明するための図である。仮想アドレス空間 9 2 0 は、CPU のメモリマップで、例えば 0 番地から、内蔵メモリ、外部空間 0、外部空間 1 などいくつかの空間が割り当てられている。これらの空間は非圧縮領域として使う。コンパイル、リンクの処理により、仮想アドレス空間の内蔵メモリ領域 9 2 1 の一部に非圧縮対象のコードがマッピングされる。仮想アドレス空間 9 2 0 の丁度半分のアドレスから、再び、内蔵メモリ、外部空間 0、外部空間 1 などいくつかの空間が割り当てられており、これらは圧縮領域として使う。コンパイル、リンクの処理により、仮想アドレス空間の内蔵メモリ領域 9 2 2 の一部に圧縮対象のコードがマッピングされる。仮想アドレス空間の非圧縮部分 9 2 1 は、そのまま実メモリにマッピングされる。圧縮対象部分 9 2 2 は、圧縮ツールにより、アドレス変換情報 9 2 4、圧縮コードタイプ情報 9 2 5、圧縮コード 9 2 6、辞書 9 2 7 として、実メモリ上にコードが生成される。プログラムの実行時には、CPU は仮想アドレス空間をアクセスするが、圧縮コード伸長回路は、CPU アドレスの最上位ビットを見て、これが 0 であれば CPU アドレスをそのまま使って非圧縮コードをアクセスし、1 であればアドレス変換を行い、必要に応じて、アドレス変換情報、圧縮コードタイプ情報、圧縮コード、辞書を読み出し、圧縮コードの伸長を行う。

【 0 0 7 5 】

#### 【発明の効果】

上述のように、本発明によれば、マイコン等の組み込み機器に好適なコード圧縮技術が提供される。

#### 【図面の簡単な説明】

【図 1】は、本発明を適用したマイクロコントローラの実施例の構成の要部を説明するためのブロック図である。

【図 2】は、本発明の基本的な原理を示す第 1 の実施例における、圧縮コード 2 3 の 4 つのフォーマットである。(a) は、コード長が 4 ビットのタイプ 0 フォーマットである。(b) は、コード長が 8 ビットのタイプ 1 フォーマットである。(c) は、コード長が 1 0 ビットのタイプ 2 フォーマットである。(d) は、コード長が 1 6 ビットのタイプ 3 フォーマットである。



【図 3】は、第 1 の実施例における、圧縮前のコードと圧縮コードとの対応を説明するための図である。

【図 4】は、第 1 の実施例における、圧縮前のコードのブロックとアドレス変換情報のアドレスの対応を説明するための図である。

【図 5】は、第 1 の実施例における、アドレス変換情報のメモリアドレスの計算方法を示す図である。

【図 6】は、第 1 の実施例における、圧縮コード 1 ブロックに対する圧縮コードタイプ情報を示す図である。

【図 7】は、第 1 の実施例における、圧縮前のコードと圧縮コードタイプ情報のアドレスの対応を説明するための図である。

【図 8】は、第 1 の実施例における、圧縮コードタイプ情報のメモリアドレスの計算方法を示す図である。

【図 9】は、第 1 の実施例における、辞書のメモリマップを示す図である。

【図 1 0】は、第 1 の実施例における、辞書のメモリアドレスの計算方法を示す図である。

【図 1 1】は、第 1 の実施例における、ブロックの先頭から目的の圧縮コードまでのビット数を計算する回路の例を示す図である。

【図 1 2】は、第 1 の実施例における、ブロックの先頭から目的の圧縮コードまでのビット数を計算する回路において、圧縮コードタイプ情報を圧縮コードサイズに変換する回路 3 3 0 の動作を示す図である。

【図 1 3】は、第 1 の実施例における、ブロックの先頭から目的の圧縮コードまでのビット数を計算する回路において、1 ブロックの圧縮コードサイズを目的の圧縮コードに対応する部分とこれ以降を 0 にマスクする回路 3 3 1 の動作を示す図である。

【図 1 4】は、第 1 の実施例における、ブロックの先頭から目的の圧縮コードまでのビット数を計算する回路において、マスク後の圧縮コードサイズを加算し、圧縮コードのブロック内アドレスを計算する回路 3 3 2 のブロック図である。

【図 1 5】は、第 1 の実施例における、回路 3 3 2 の構成要素の 1 つである、桁上げ保存加算アレイ 3 3 2 - 1 の例である。

【図 1 6】は、第 1 の実施例における、桁上げ保存加算アレイ 3 3 2 - 1 の構成要素である、桁上げ保存加算器の動作を示す図である。

【図 1 7】は、第 1 の実施例における、圧縮コードのアドレス計算方法を示す図である。

【図 1 8】は、第 1 の実施例における、メモリより読み出した圧縮コードから、目的の圧縮コードを抽出する方法を示す図である。

【図 1 9】は、第 1 の実施例における、圧縮コードの伸長フローを示す図である。

【図 2 0】は、本発明を適用したマイクロコントローラの第 2 の実施例の構成の要部を説明するためのブロック図である。

【図 2 1】は、本発明を適用したマイクロコントローラの第 3 の実施例の構成の要部を説明するためのブロック図である。

【図 2 2】は、本発明を適用したマイクロコントローラの第 4 の実施例の構成の要部を説明するためのブロック図である。

【図 2 3】は、本発明におけるマイコンプログラムの圧縮の流れを示す図である。

【図 2 4】は、本発明における非圧縮コードと圧縮コードを混在させる方法を説明するための図である。

【符号の説明】

1 … C P U

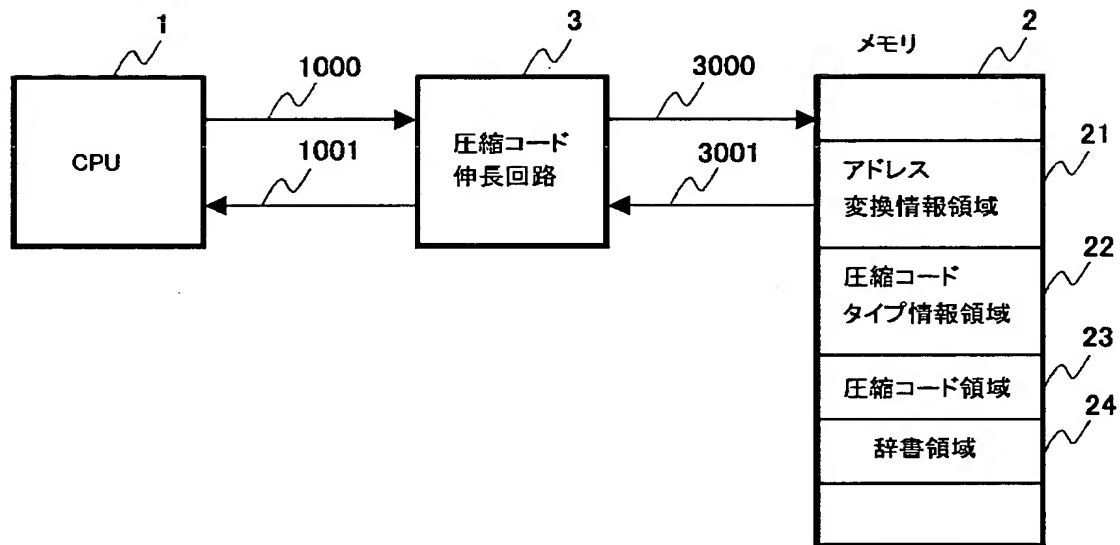
2 … メモリ

3 … 圧縮コード伸長回路

【書類名】 図面

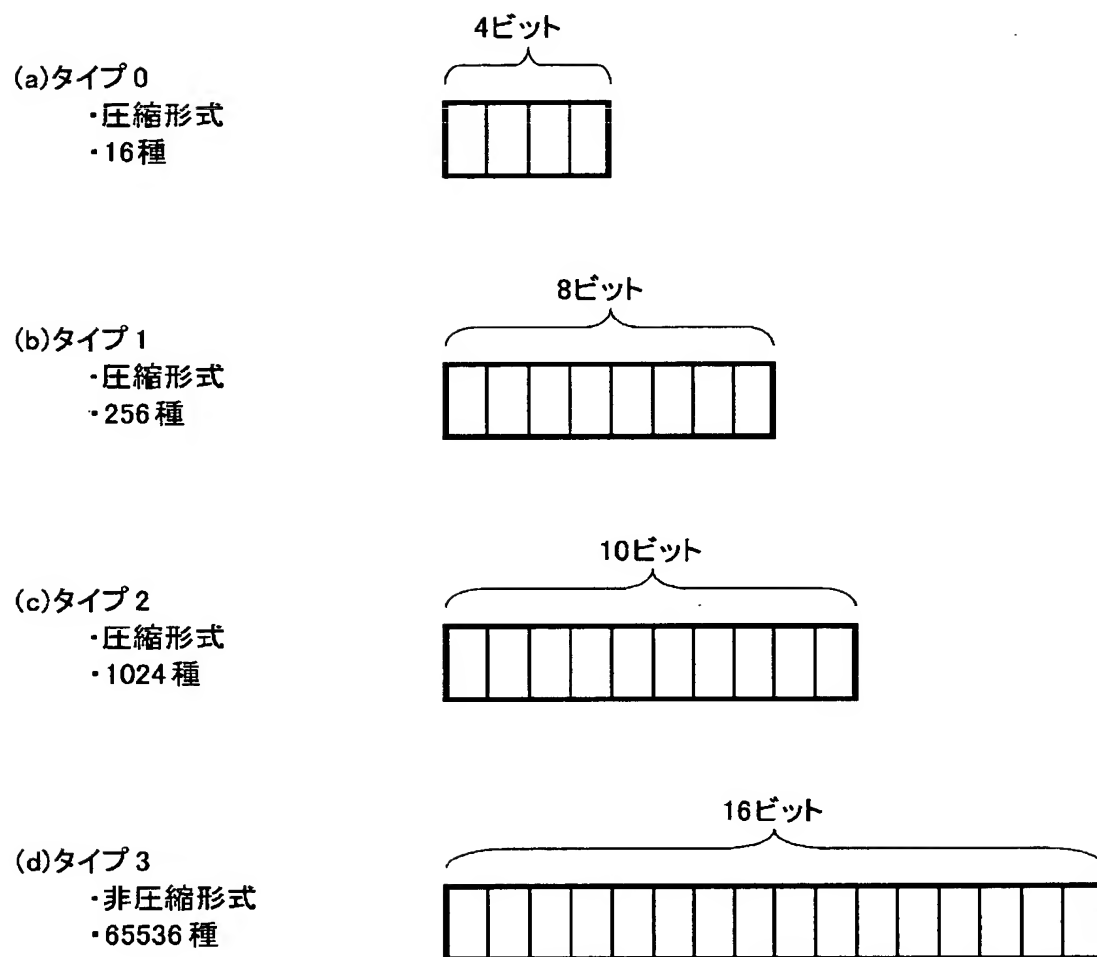
【図 1】

図 1



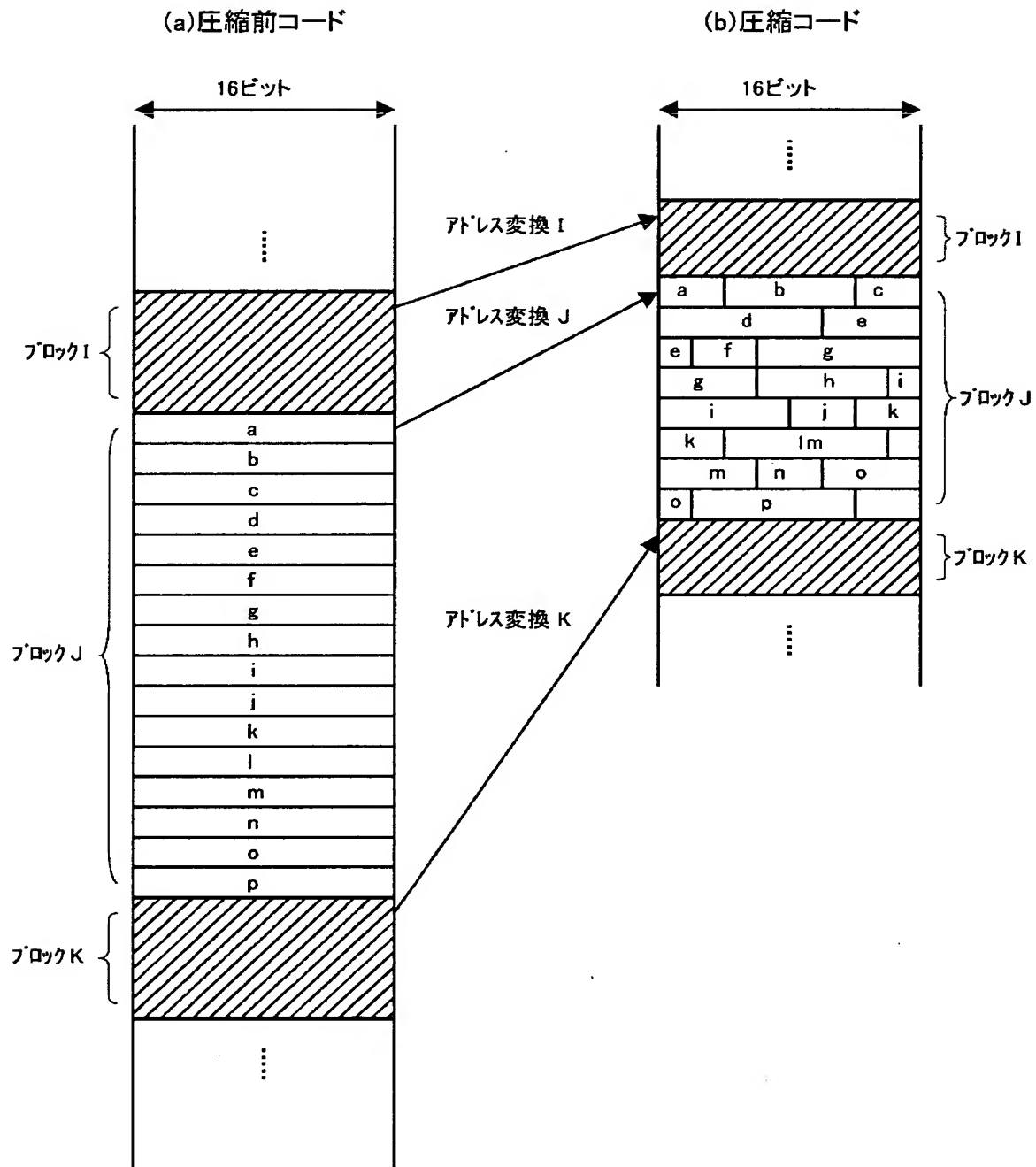
【図 2】

図2



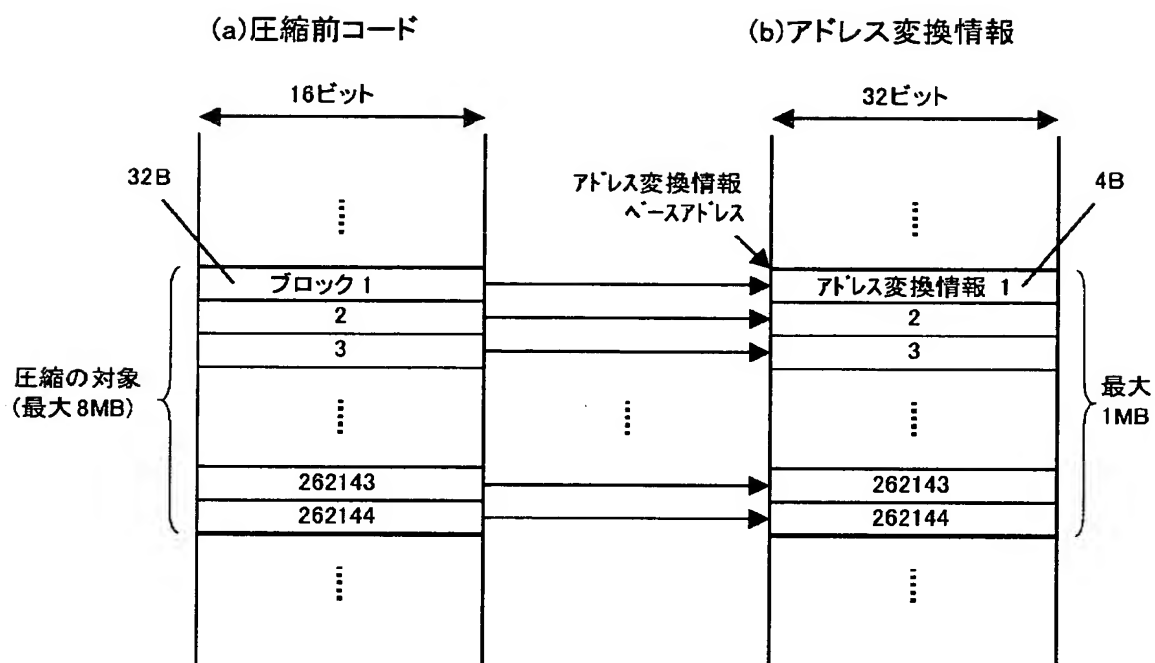
【図 3】

図 3



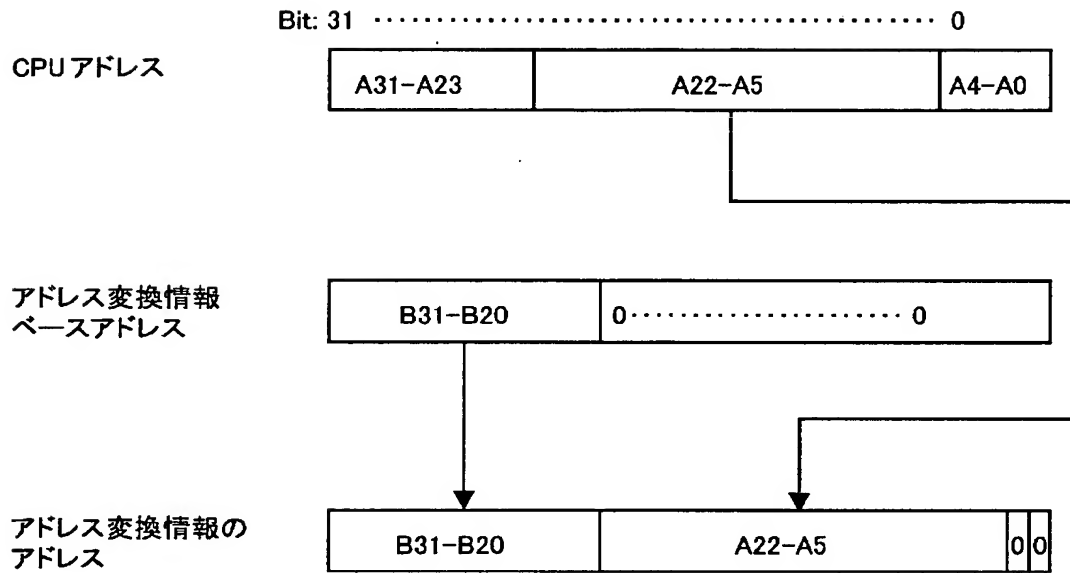
【図 4】

図 4



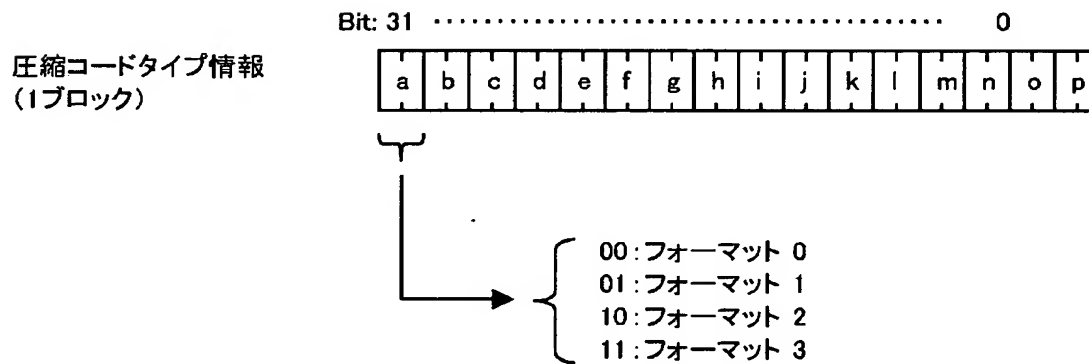
【図 5】

図5



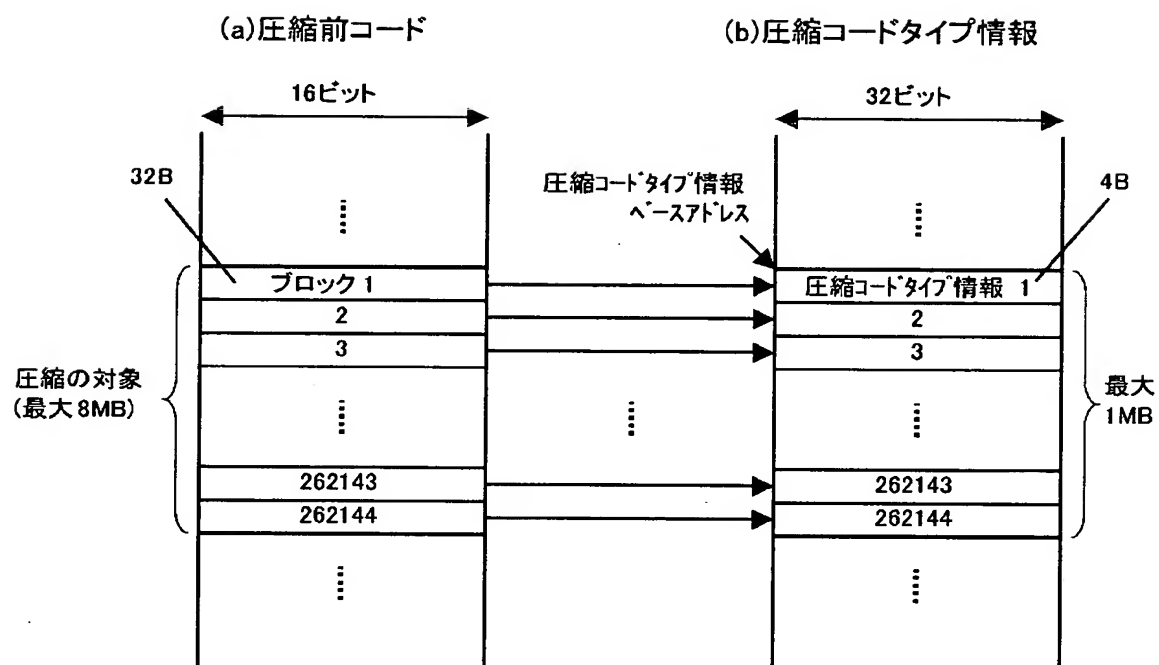
【図 6】

図6



【図 7】

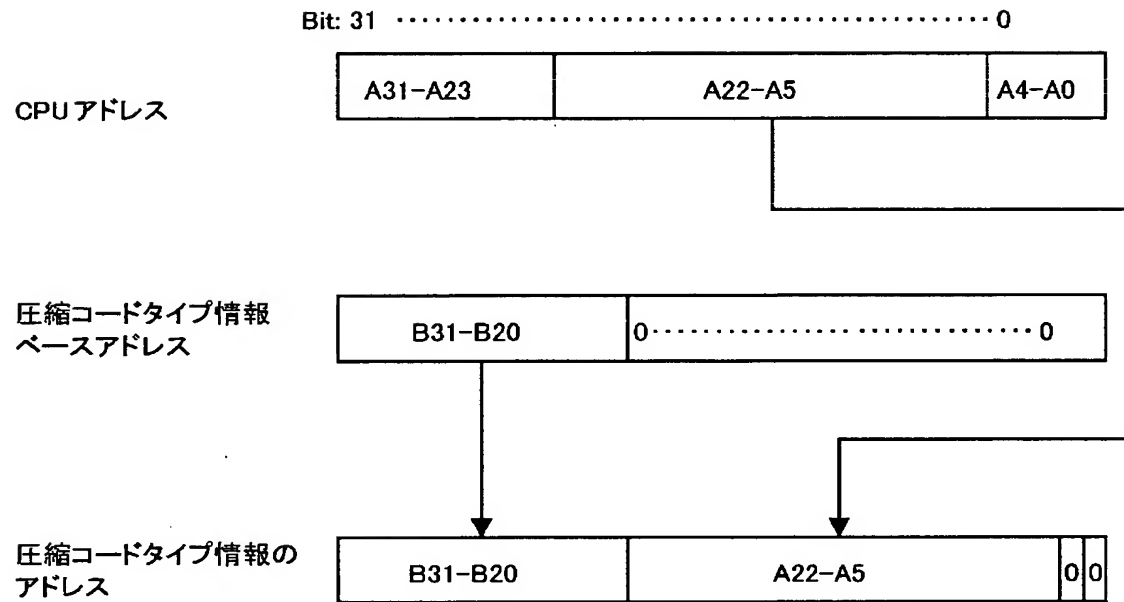
図 7





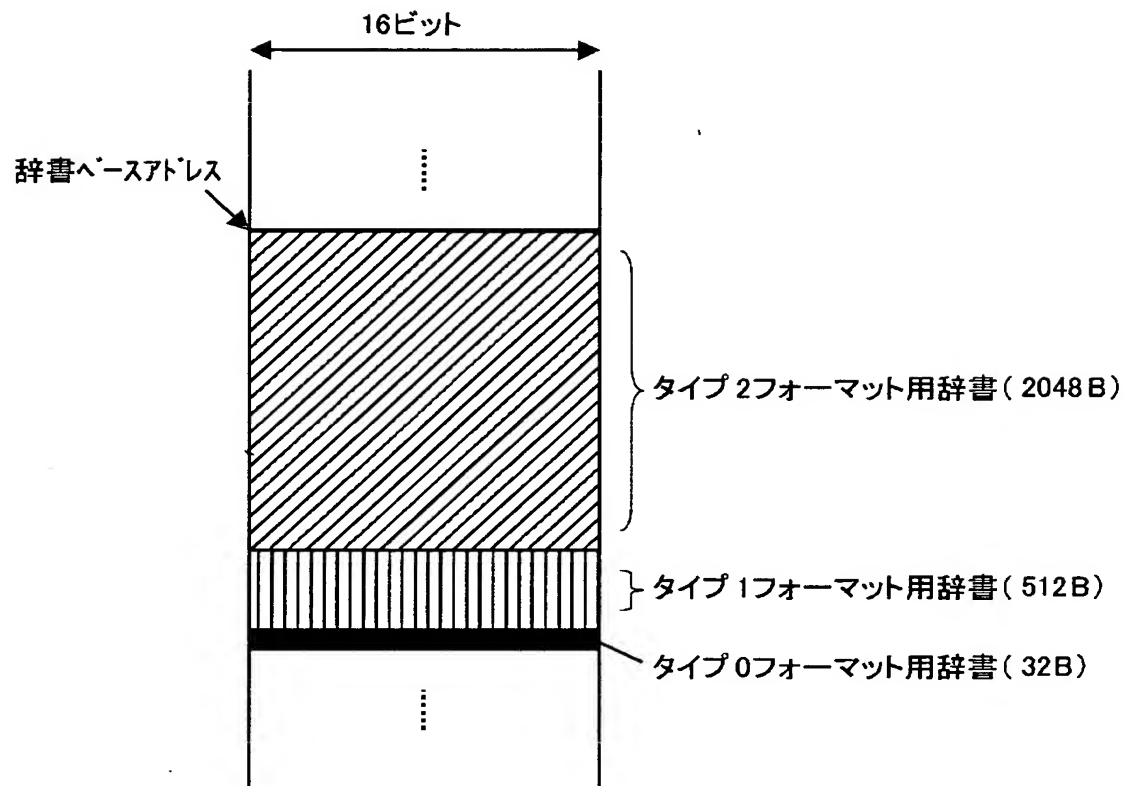
【図 8】

図 8



【図 9】

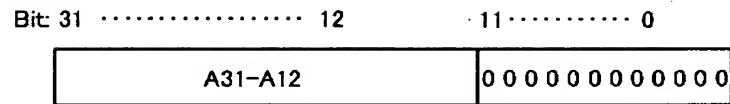
図9



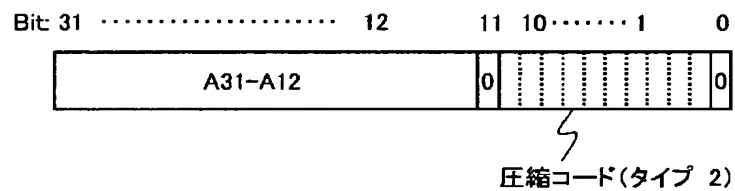
【図 1 0】

図 10

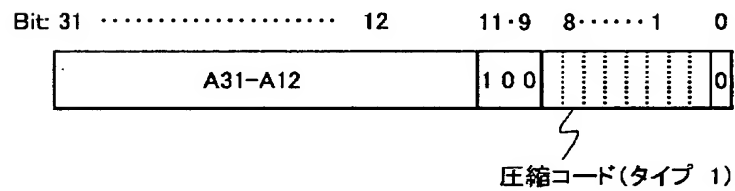
(a)辞書ベースアドレス



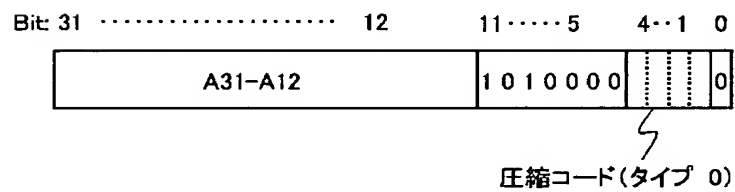
(b)辞書(タイプ 2)アドレス



(c)辞書(タイプ 1)アドレス

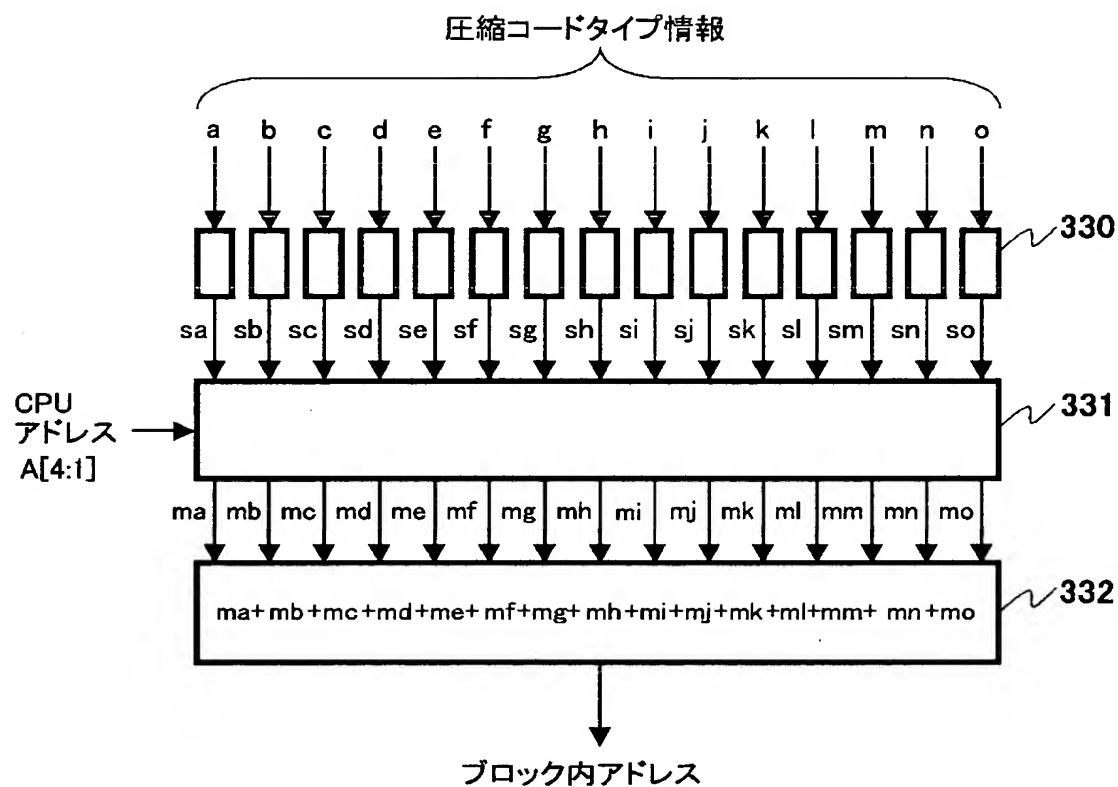


(d)辞書(タイプ 0)アドレス



【図 1 1】

図 1 1



【図 1 2】

図 1 2

入力 i	出力 o
00	0010
01	0100
10	0101
11	1000

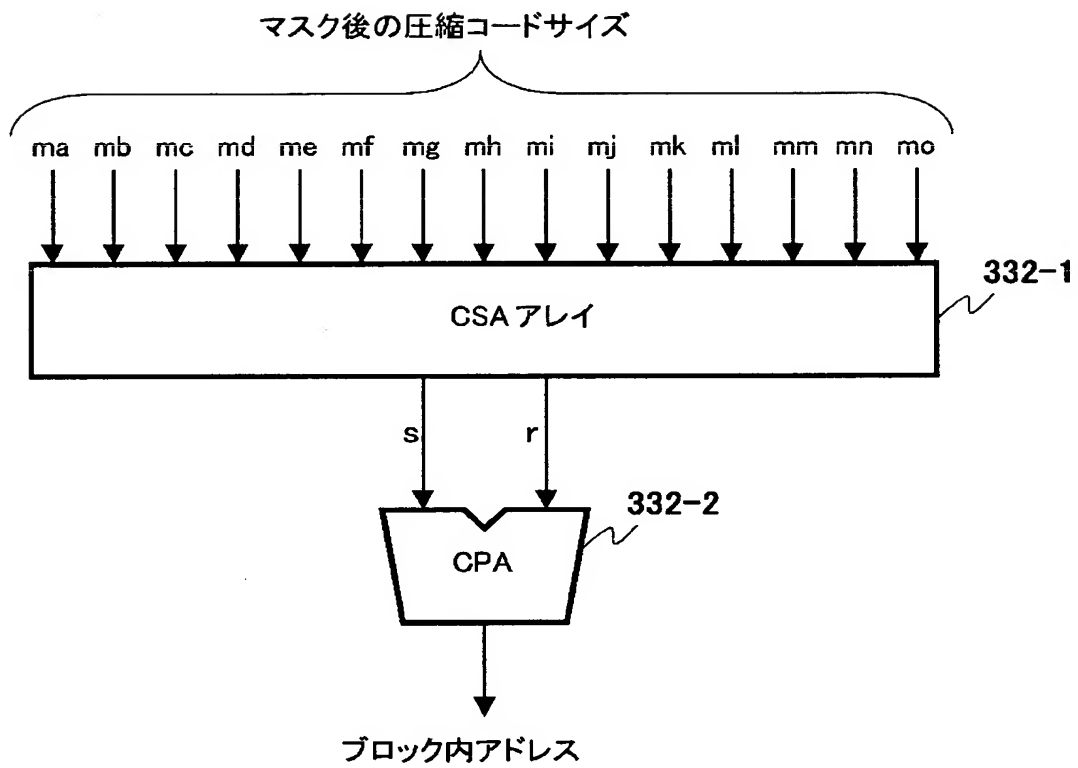
【図 1 3】

図13

A[4:1]	ma	mb	mc	md	me	mf	mg	mh	mi	mj	mk	ml	mm	mn	mo
0000	Z	Z	Z	Z	Z	Z	Z	Z	Z	Z	Z	Z	Z	Z	Z
0001	sa	Z	Z	Z	Z	Z	Z	Z	Z	Z	Z	Z	Z	Z	Z
0010	sa	sb	Z	Z	Z	Z	Z	Z	Z	Z	Z	Z	Z	Z	Z
0011	sa	sb	sc	Z	Z	Z	Z	Z	Z	Z	Z	Z	Z	Z	Z
0100	sa	sb	sc	sd	Z	Z	Z	Z	Z	Z	Z	Z	Z	Z	Z
0101	sa	sb	sc	sd	se	Z	Z	Z	Z	Z	Z	Z	Z	Z	Z
0110	sa	sb	sc	sd	se	sf	Z	Z	Z	Z	Z	Z	Z	Z	Z
0111	sa	sb	sc	sd	se	sf	sg	Z	Z	Z	Z	Z	Z	Z	Z
1000	sa	sb	sc	sd	se	sf	sg	sh	Z	Z	Z	Z	Z	Z	Z
1001	sa	sb	sc	sd	se	sf	sg	sh	si	Z	Z	Z	Z	Z	Z
1010	sa	sb	sc	sd	se	sf	sg	sh	si	sj	Z	Z	Z	Z	Z
1011	sa	sb	sc	sd	se	sf	sg	sh	si	sj	sk	Z	Z	Z	Z
1100	sa	sb	sc	sd	se	sf	sg	sh	si	sj	sk	sl	Z	Z	Z
1101	sa	sb	sc	sd	se	sf	sg	sh	si	sj	sk	sl	sm	Z	
1110	sa	sb	sc	sd	se	sf	sg	sh	si	sj	sk	sl	sm	sn	
1111	sa	sb	sc	sd	se	sf	sg	sh	si	sj	sk	sl	sm	sn	so

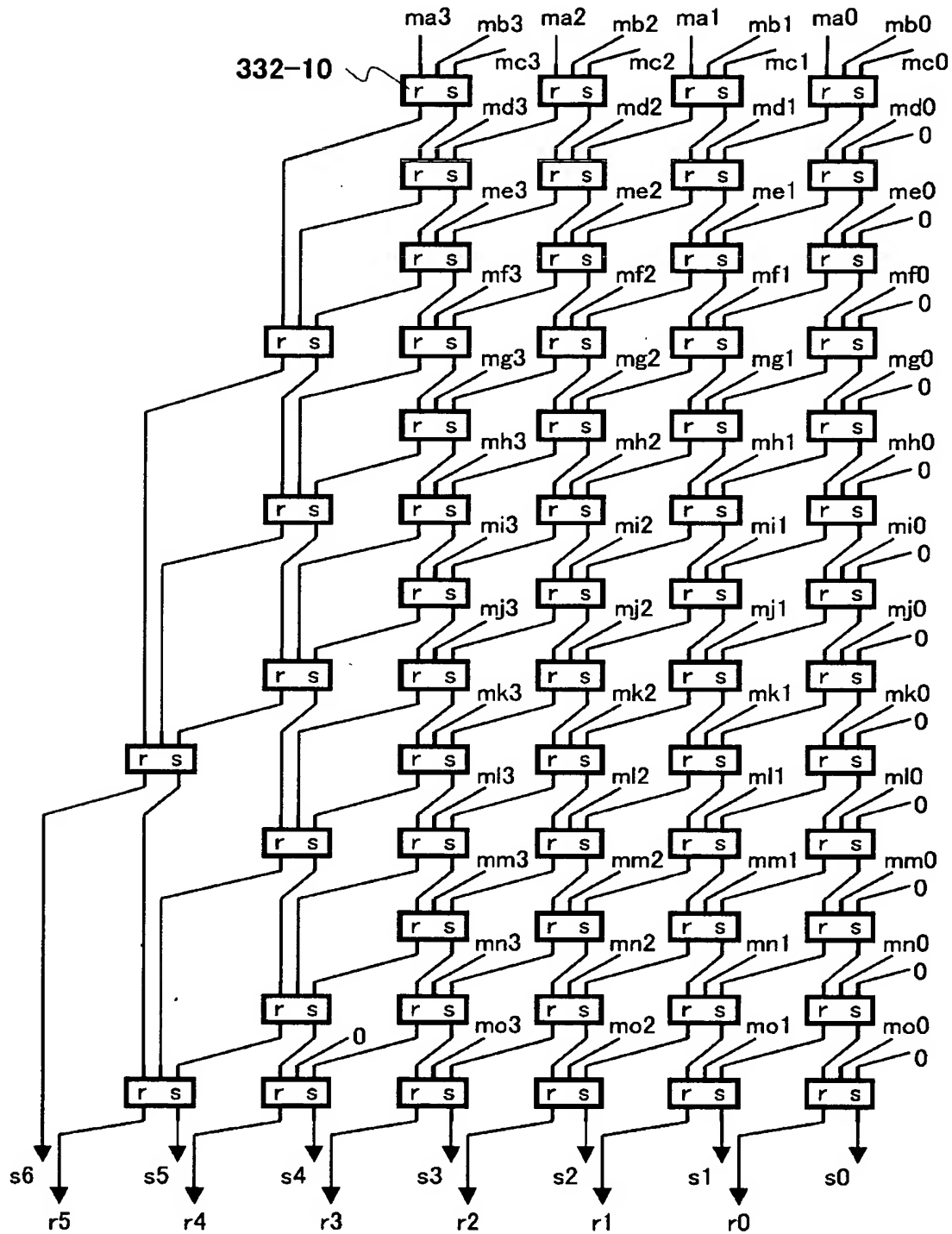
【図 1 4】

図 1 4



【図15】

図15



【図 1 6】

図16

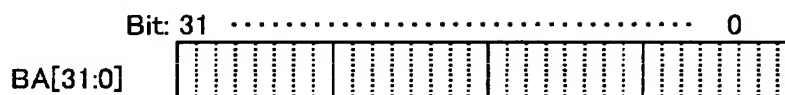
i0	i1	i2	r	s
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1



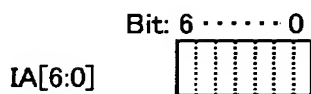
【図 1 7】

図 17

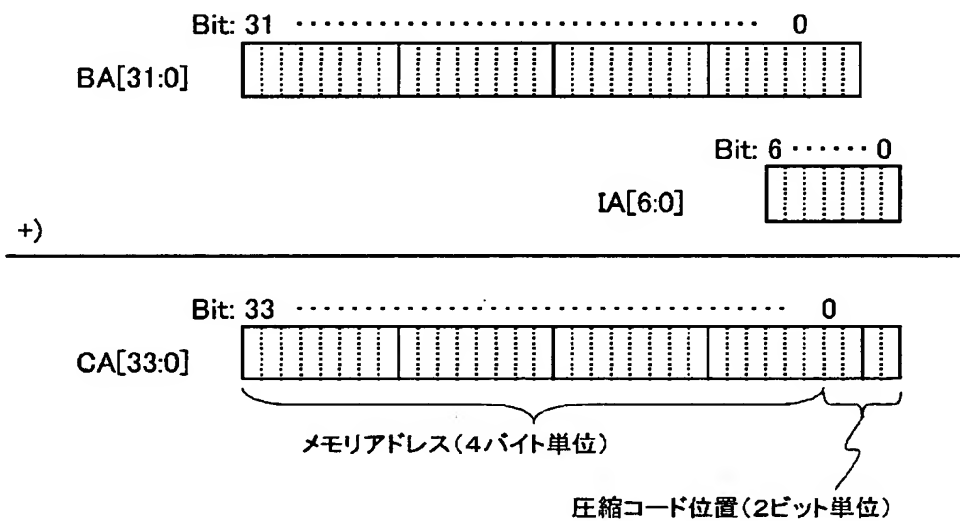
(a) アドレス変換情報(圧縮コードブロック先頭アドレス)



(b) ブロック内アドレス

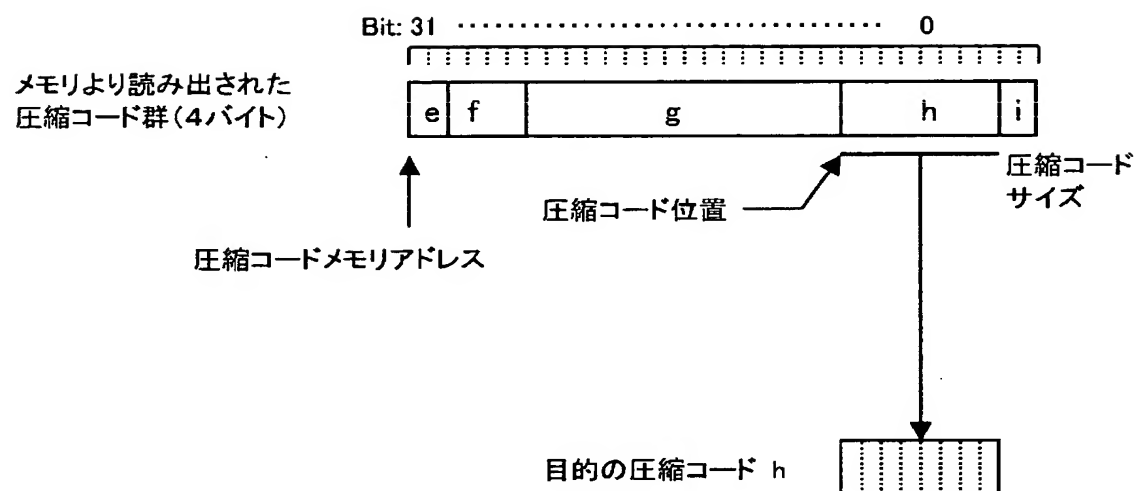


(c) 圧縮コードアドレス



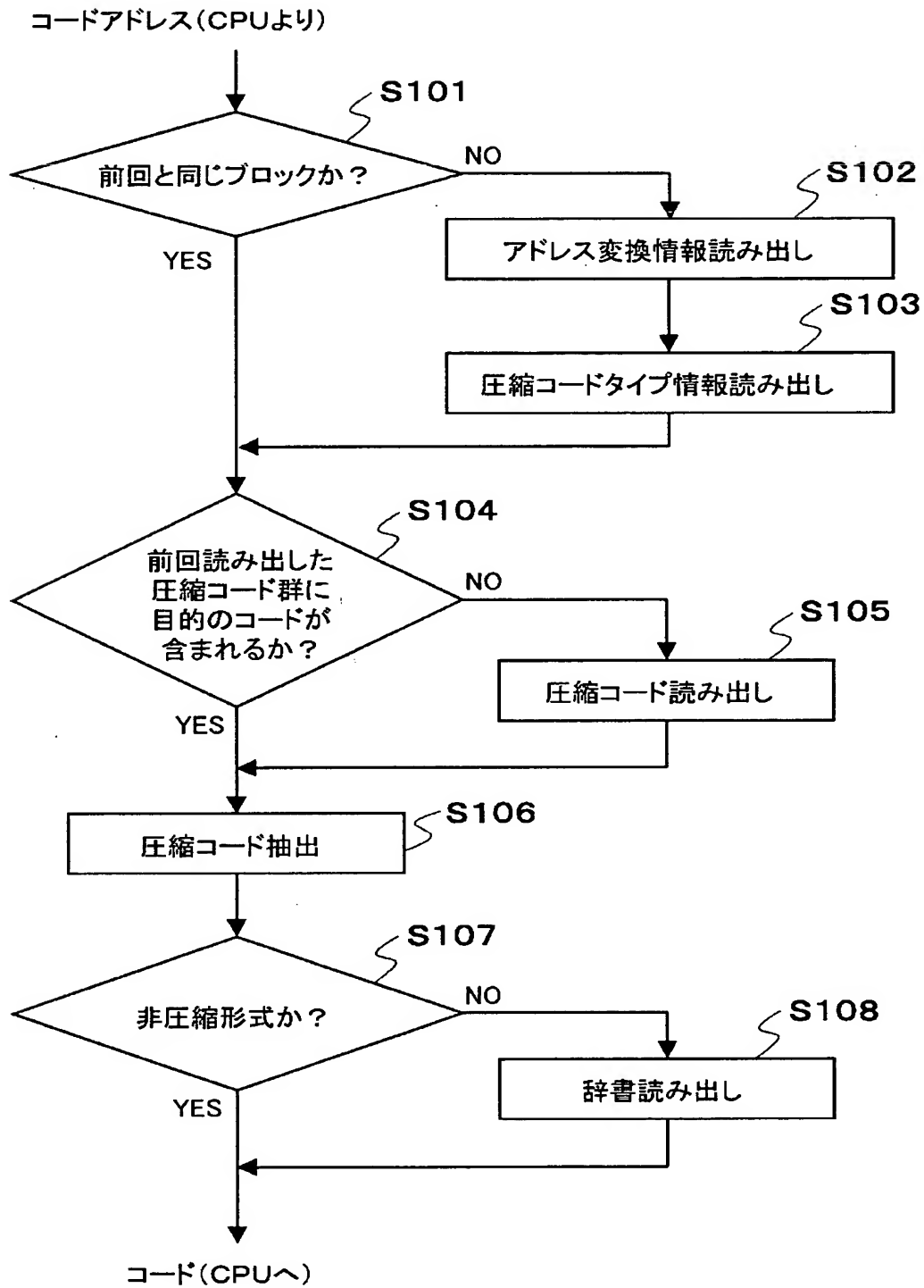
【図 1 8】

図18



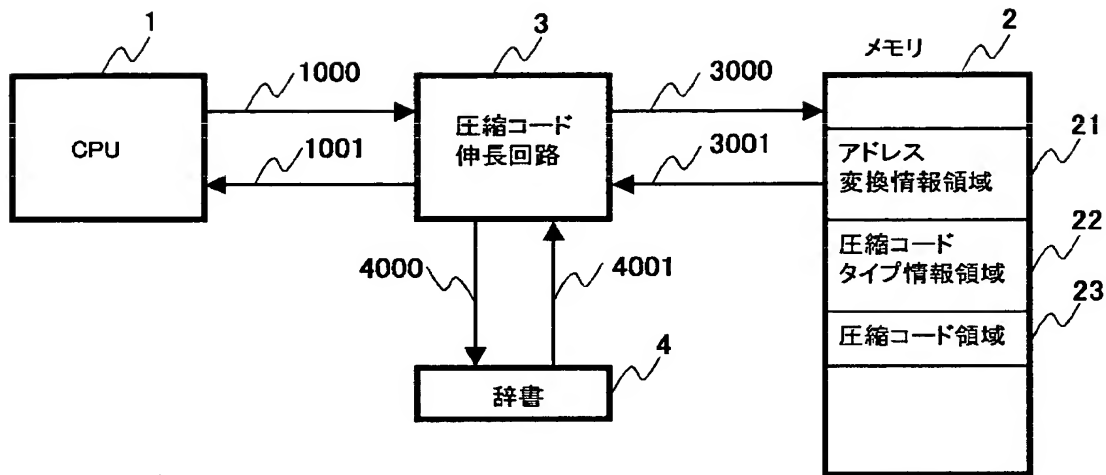
【図 1 9】

図19



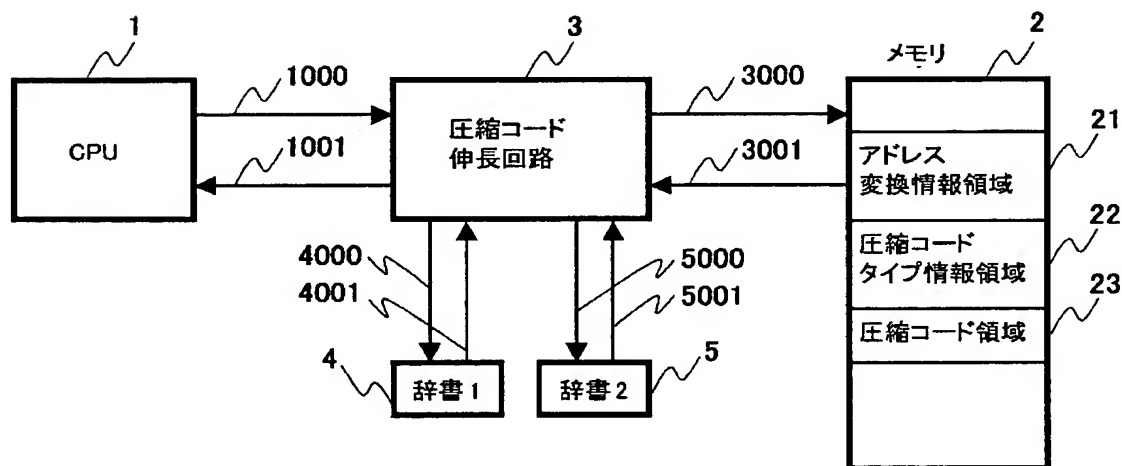
【図 2 0】

図20



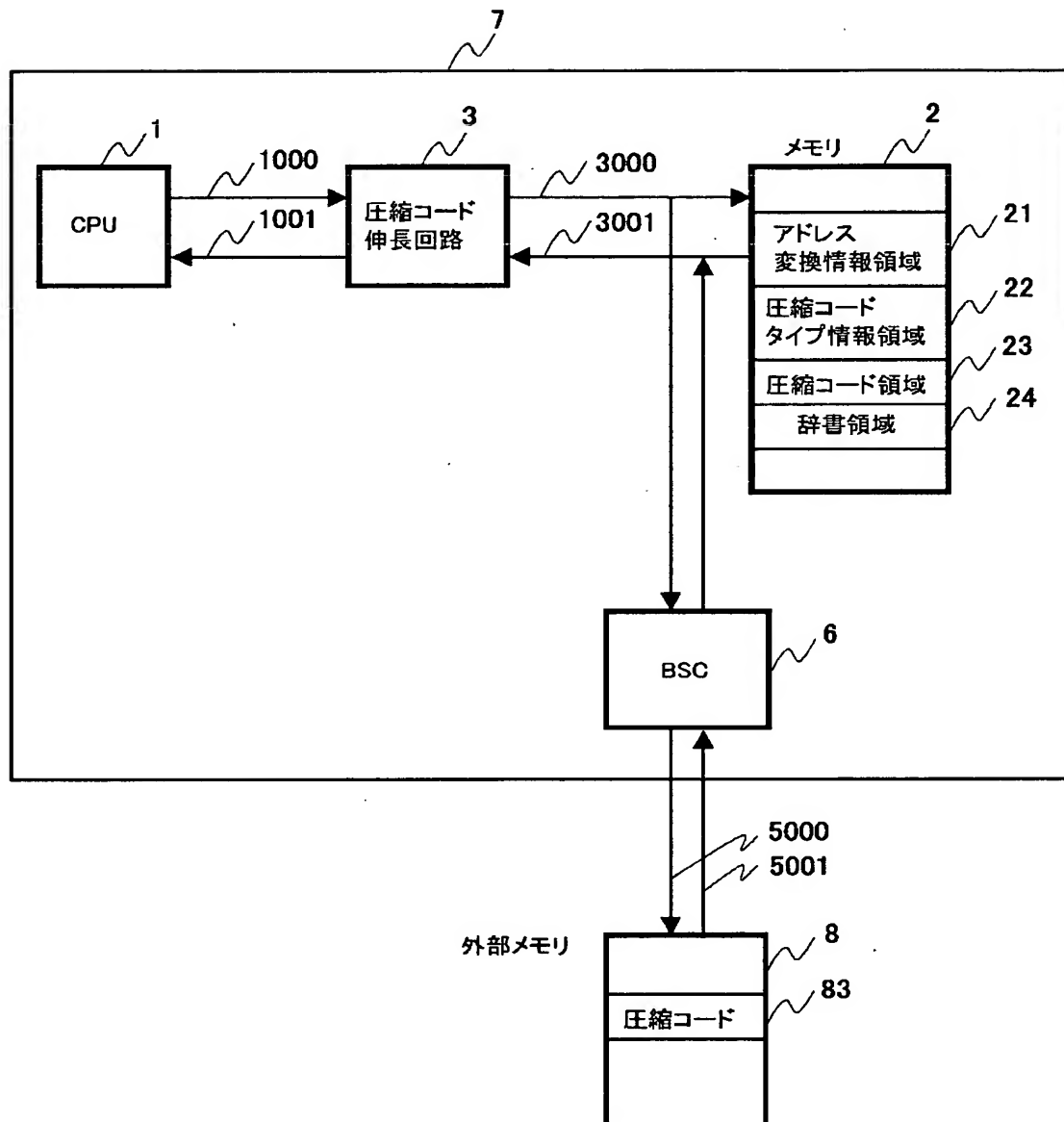
【図 2 1】

図21



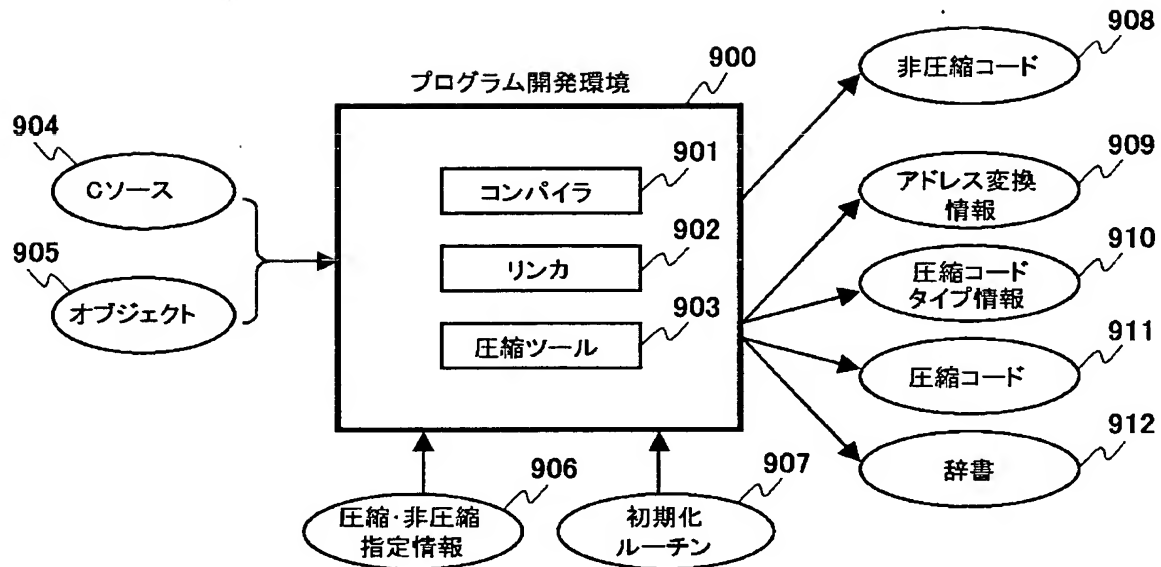
【図 22】

図22



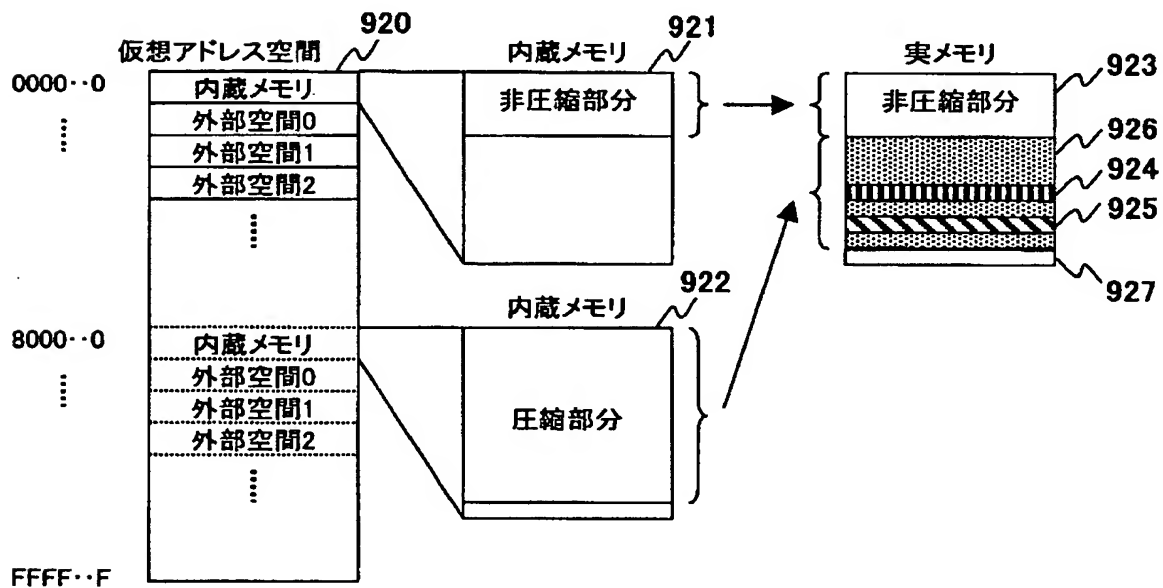
【図23】

図23



【図24】

図24



【書類名】 要約書

【要約】

【課題】 マイコン等の組み込み機器に好適なコード圧縮技術を提供する。

【解決手段】 プログラムのコードを可変長コードに変換した圧縮コードと、グループ化されたプログラムのコードについて、各グループの開始アドレスを特定するためのアドレス変換情報と、各グループごとに、グループに含まれる各圧縮コードのコード長を特定するための圧縮コードタイプ情報とをメモリに記憶し、CPUが出力するコードアドレスから対応する圧縮コードアドレスを直接計算できるように構成することで、マイコン等の組み込み機器に好適なコード圧縮が達成される。

【選択図】 図 1

出 願 人 履 歴 情 報

識別番号 [ 0 0 0 0 0 5 1 0 8 ]

1. 変更年月日	1 9 9 0 年 8 月 3 1 日
[変更理由]	新規登録
住 所	東京都千代田区神田駿河台 4 丁目 6 番地
氏 名	株式会社日立製作所